REMARKS/ARGUMENTS

Claims have been amended to further clarify the subject matter regarded as the invention. No new matter has been added. The clarifications made are, for example, supported by the Specification, page 8, line 18 to page 9, line 26.

Claims 15-20 have been canceled. New claims 21-26 have been added. Thus, claims 1-14 and 21-26 are now pending.

In the Office Action, the Examiner rejected claims 1-14 under 35 U.S.C. §102(e) as being anticipated by U.S. Patent No. 6,481,006 (*Blandy et al.*) This rejection is fully traversed below.

The present application relates to techniques for invocation of native methods in Java computing environments (i.e., invoking a native method from a Java computing environment). These techniques can be implemented in Java computing environments to facilitate use of methods (functions or subroutines) written in programming languages other than Java (e.g., C, C++, etc.). In accordance with one aspect of the invention, a lightweight native method invocation interface is disclosed. In one embodiment, the lightweight native method invocation provides direct access to Java parameters on a Java execution stack. In addition, the lightweight native method invocation can include macro instructions that use the reference to generate native parameters. It should be noted that Java parameters can be converted into native parameters if it is necessary to do so. The lightweight native method invocation can significantly reduce the overhead associated with conventional techniques because, among other things, there is no need to use a conventional Java Native Interface (JNI) to invoke native methods. As a result, performance of virtual machines, especially those operating with relatively less memory and/or computing power, can be improved (see, for example, Specification, page 3, lines 3-20).

By way of example, claim 1 pertains to a method of invoking a native method written in a programming language other than Java. As such, claim 1 recites (a) providing a reference to one or more Java parameters on a Java execution stack. It should be noted that the one or more Java parameters are parameters associated with a native method that is to be invoked. Claim 1 also recites: (b) generating one or more native parameters based on the reference, and (c) invoking the native method with the one or more native parameters.

In contrast, *Blandy et al.* teaches invocation of Java methods from native code (*Blandy et al.*, Abstract). As such, *Blandy et al.* does NOT teach or suggest techniques for invocation of native methods from Java. This is believed to be evident because *Blandy et al.* does not address invocation of native methods from Java. Instead, *Blandy et al.* addresses the reverse process,

namely, invocation of Java methods from native code. Accordingly, it is respectfully submitted that *Blandy et al.* does NOT teach or suggest (a) providing a reference to one or more Java parameters relating to a native method, that are stored on a Java execution stack. It should be noted that the one or more parameters stored on the Java stack are Java parameters that are used to invoke a native method (i.e., parameters that are passed to the native method, for example, as input). The reference that is provided is used to generate one or more native parameters. These native parameters are, in turn, used as parameters for invoking the native method when the native method is invoked. (e.g., native parameters that are placed on a native execution stack when the native method is invoked). *Blandy et al.* also fails to teach or suggest (b) generating the native parameters, and (c) invoking the native method using the native parameters.

Claim 1 recites all of these features (a, b, and c). Accordingly, it is respectfully submitted that claim 1 is patentable over *Blandy et al.* for these reasons. In addition, claims that are dependent on claim 1 are also patentable over *Blandy et al.* for at least these reasons alone. Moreover, these dependent claims recite additional features that render them patentable for additional reasons. For example, claim 5 additionally recites converting at least one of the Java Parameters to a native parameter.

In the Office Action, the Examiner has asserted that *Blandy et al.* teaches this feature (Office Action, page 3, citing Col. 5, lines 24-25, of *Blandy et al.*) It is noted that *Blandy et al.* states that parameters from a native stack are copied to a Java stack (*Blandy et al.*, Col. 5, lines 6-8). It is also noted that *Blandy et al.* further describes that a parameter that exists on the native stack as a 64-bit floating point is converted to a 32-bit floating point value before being copied to the Java stack (*Blandy et al.*, Col. 5, lines 22-26).

Accordingly, *Blandy et al.* describes converting native parameters to Java parameters. Thus, contrary to the Examiner's assertion, *Blandy et al.* does NOT teach converting Java parameters into native parameters. Instead, conversion of native parameter to Java Parameters are used by the methodology taught by *Blandy et al.* to facilitate invocation of Java methods from native codes. This is also evident because *Blandy et al.* pertains to invocation of Java methods from native code and NOT invocation of native methods from Java. Thus, claim 5 is patentable over *Blandy et al.* for additional reasons.

Independent claim 10, among other things, recites providing a reference to Java parameters associated with a native method, and using the reference to convert them into native. Accordingly, it is respectfully submitted that claim 10 and its dependent claims are also patentable over *Blandy et al.* for similar reasons as discussed above. It should also be note that the new claims also recite similar features as discussed above.

Based on the foregoing, it is submitted that claims are patentably distinct over the cited art of record. Additional limitations recited in the independent claims or the dependent claims are not further discussed because the limitations discussed above are sufficient to distinguish the claimed invention from the cited art. Accordingly, Applicant believes that all pending claims are allowable and respectfully requests a Notice of Allowance for this application from the Examiner.

Applicants hereby petition for an extension of time which may be required to maintain the pendency of this case, and any required fee for such extension or any further fee required in connection with the filing of this Amendment is to be charged to Deposit Account No. 500388 (Order No. SUN1P829). Should the Examiner believe that a telephone conference would expedite the prosecution of this application, the undersigned can be reached at the telephone number set out below.

Respectfully submitted,
BEYER WEAVER & THOMAS, LLP

R. Mahboubian
Reg. No. 44,890

P.O. Box 778
Berkeley, CA 94704-0778
(650) 961-8300